1. Bus

Program Name: Bus.java **Input File: bus.dat**

The City of Silliness has purchased a fleet of new busses that are so skinny that people must stand in them single file and cannot walk around each other.

- The busses have two doors, one in the very front by the driver, and one in the very back.
- New passengers must enter through the front to pay their fare, but can exit the bus via the front or the back door.
- Passengers always enter the bus in the order listed. When passengers exit the bus to let other passengers exit, the re-entering passengers will always enter in the reverse order they exited so that their original order is maintained.
- Assume that everyone on the bus will want to inconvenience the fewest number of people when they exit.
    - A passenger can exit either the front door or the back door, whichever is closest to them.
    - If somebody needs to exit the bus that isn't the first or last person then people in front or behind that person will need to get off the bus to allow him or her to exit, then they must get back on to continue the trip to their stops. Re-entering passengers must maintain their original order.
    - If more than one passenger needs to exit at a stop, they can all exit the front door or the back door, or some can exit the front and others the back, whichever is most convenient for the other passengers. Then the other passengers will re-enter the bus, maintaining their original order.
- While this method is very tedious for people on the bus, this new model saves the city a lot of money!

The city is interested to know how many people walk in and out of the doors on a given bus route, whether they are getting on the bus for the first time, getting off the bus at their exit, or having to exit then re-enter because somebody needs to leave.

- A person getting on the bus counts as 1 enter/exit, and a person getting off at their stop also counts as 1.
- If somebody has to get off and then back in to let somebody out, both the exit and enter are counted, so the value is incremented by 2.

Input

The input file will contain an unknown number of lines:

- The file will not contain any blank lines.
- The file may contain multiple routes.
- Each line will be of the form:
    - `ON` followed by a space and a space-delimited list of upper case ASCII characters denoting the people getting on the bus at a particular point in the route.
    - `OFF` followed by a space and a space-delimited list of upper case ASCII characters denoting the people getting off the bus at a particular point in the route (these people are guaranteed to already be on the bus).
    - `END ROUTE` – This means the current route is ended, any remaining people on the bus must exit, and the data should be calculated and printed out.
- The last line is guaranteed to be `END ROUTE` so that no bus route is left in limbo.

Output

For each route you will print a single line containing a string in the form `People Entered/Exited: N`, where N is the number of exits/enters during the given route.

Example Input File

```
ON A B C
OFF C
ON D E F G
OFF E B
END ROUTE
```

Example Output to Screen

```
People Entered/Exited: 18
```